REMARKS

Known linkers report errors and may fail to complete the allocation of ingredients object files if there are unresolved symbolic references. The user then re-edits to improve or adjust the linking instructions. This activity inhibits interactive allocation strategies in which a user attempts to optimize the allocation of only a part of the ingredients of the software program before the remaining parts are available or written. No links may be left incomplete. Therefore, extensive experimentation is prohibited and users are discouraged from finding more optimal ways of linking. Known linkers are unable to resolve incomplete links. All sections must fit in memories before an output file may be created or the symbol references resolved. Unlike general purpose processor having a single, large memory, embedded processors have many different memories. The layout of the particular application into different target memories impacts performance. Certain kinds of fast memory, such as on-chip memory, are limited in space and desired fro critical application functions. Trade-offs is made depending on the size of the programmer's application plus any third party components and libraries. As the program evolves and grows, the allocation decisions are revised in a time consuming manner. It is desirable to provide a system that allows a user, software program or component via command commands, gestures, and

application programming interfaces to specify a link in an incomplete fashion. During such incremental specification, the link may be incomplete. The ingredient object files need not be complete and not all sections from each object file need not be placed or allocated into memory. The symbolic references need not be resolved. It is desirable to have a linker that communicates visually to the user without producing the errors or other procedures to prevent further linking operations. The feedback desired is the allocated position and size of the sections that are allocated to memory, the values of the symbols that are allocated, a list of symbolic references that are not defined, a list of ingredient object files and sections that are not allocated. The user receives real-time on the results of the linking operations even if not all files and sections are allocated to a location in memory. The user may therefore experiment with different linking strategies without the need for actually completing the link.

Claims 1-9, 12 and 13 are rejected under 35 U.S.C. 103 (a) as being unpatentable over Lawrence et al. (U.S. Patent No. 5,519,866) hereinafter Lawrence) further in view of McLain, Jr. (U.S. Patent No. 5,956,513) (hereinafter Mc Lain). Applicant's claimed invention in amended claim 1 calls for "an allocation module for allocating sections of code and data into different target memories of a processor without running a confidence

check" and "an incomplete link module, wherein said incomplete link comprises allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated without running a confidence check and without actually completing the link; said allocation information including the allocated position and size of those sections that are allocated to said different target memories." As noted by the examiner Lawrence does not teach or suggest an incomplete Link module that comprises allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated. The examiner references McLain. The ABC in McLain builds an internal table that lists each program file, object module and header file, along with its relevant time stamp, to inform the user when each respective file was created. The internal table refers to temporary data stored in the memory of the computer that embodies ABC, and will later be viewable in a results report. The process is performed for each program file and is repeated until table entries for all programs files are completed. During this step error messages may be generated and sent to the results report. Like wise any potential error or conflict that is detected is logged and presented to the user. In step 208 ABC performs date/time stamp comparison between the object models and header files to determine which source modules are required to be recompiled. In step 210, the ABC performs a syntax check on the commands that invoke the compiler and the linker. Step 212 determines any unresolved conflicts. If unresolved it simply prompts the user to

proceed or terminate. If not resolved the unresolved conflicts are presented to the user.

Mc Lain does not have an allocation module for allocating sections of code and data into different target memories of a processor without running a confidence check. It is not seen where different target memories is discussed in McLain. Step 212 in Fig. 2 of McLain is a confidence check. Step 214 presents the conflicts and prompts the user to proceed or terminate. Mc Lain does not have an incomplete link module, wherein said incomplete link comprises allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated perform without running a confidence check and without actually completing the link. Further, McLain does not teach or suggest that the said allocation information includes the allocated position and size of those sections that are allocated to different memories. As discussed previously applicant's incomplete link without running confidence checks allows the user to specify a link in an incomplete fashion so the user may experiment with different linking strategies without the need for actually completing the link. It also allows the user to view the memory layouts in multiple and_different target memories. If the user in McLain elects to proceed with an unresolved conflict the compiler and linker are invoked and the link is completed. There is no suggestion of an incomplete link module as claimed or allocation without actually completing the link. It is not seen where there is anything about position and size allocated to memory in McLain or to different target

memories. Nothing like this is taught in Lawrence either. Claim 1 further calls for a link server that interprets linking instructions modifies such an incomplete link module accordingly and an interface for receiving instructions for said link server and for providing feedback as to the state of this incomplete link. This is not taught in either Lawrence or McLain reference. There no such incomplete link as claimed or means for providing feedback as to the status of this incomplete link where allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated perform without running confidence checks and without actually completing the link or said allocation information includes the allocated position and size of those sections that are allocated to different target memories.

Claim 1 further calls for a graphical user interface that generates said instructions in response to user gestures and graphically displays the state of said incomplete link with allocated and size information on those sections that are allocated to memory, the values of symbols that are allocated by said allocation module and those that have not yet been allocated is not taught in the references. The status of such incomplete link information is not taught in McLain. Applicant's claim 1 is therefore deemed allowable over the references.

Claims 16 and 17 dependent on claim 1 are deemed allowable for at least the same reasons as claim 1. Claim 16 further

calls for "said link server and a link recipe store linking instructions received that describe how the visual linker is to be controlled and can be replayed without interaction to obtain the same effect as the sequence of commands or gestures." The examiner in rejecting claim 7 references Fig. 12 and Col. 19 lines 59-65 of Lawrence and in particular to the statement that any changes made to the Project while in the browser is automatically saved. There is no suggestion in this or seen elsewhere of Lawrence of recording linking instructions that describe how the visual linker is to be controlled or any suggestion of storing so link instructions can be replayed without interaction. Claim 17 further calls for "said stored linking instruction are displayed and altered through said graphical user interface." This is not taught or suggested in any the cited references.

Claim 2 calls for ``recording linking instructions received that describe how the visual linker is to be controlled and so that said linking instructions can be replayed without interaction to obtain the same linking effect''. It is not seen where there is any teaching or suggestion of this in Lawrence or McLain. Claim 2 includes the limitation of cancelled claim 7. The examiner in rejecting claim 7 references Fig. 12 and Col. 19 lines 59-65 of Lawrence and in particular to the statement that any changes made to the Project while in the browser is automatically saved. There is no teaching or suggestion in

Lawrence of recording linking instructions that describe how the visual linker is to be controlled or any suggestion of storing so the instruction may be replayed without interaction. It is not seen where McLain suggests this either. Claim 2 is therefore deemed allowable over the references.

Claims 3 through 6 and 8 through 13 dependent on claim 2 are deemed allowable for at least the same reasons as Claim 2. Claim 8 further calls for ``the record of link instructions may be displayed and altered through a graphical user interface.'' This is not taught in the references.

Claims 14 and 15 dependent on claim 2 are deemed allowable for at least the same reasons as claim 2. Claim 14 further calls for the step of replaying said linking instructions without interaction to obtain the same linking effect. This is not taught in the references. Claim 15 further calls for the steps of displaying the record of the link instructions and altering through a graphical user interface. It is not seen where this is taught or suggested in the references.

Claim 18 calls for ``an allocation module for allocating sections of code and data into different target memories of a processor including fast on-chip memory'' and ``an incomplete link module, wherein said incomplete link comprises allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated without actually completing the link; said allocation information including the allocated position and size of those sections that are allocated to said different target memories including fast

on-chip memory." As pointed out previously it is not seen where this is taught or suggested in the references.

In the rejection of claims 10 and 11 Draves (U.S. Patent no. 5,950,221) is cited. It is not seen where Draves teaches what is missing in Lawrence and McLain.

In view of the above applicants' Claims 1-6 and 8-20 are deemed allowable over the references and an early notice of allowance of these claims is deemed in order and is respectfully requested.

Respectfully submitted,

Robert L. Troike

Robert L. Troike

Reg. No. 24183

(972) 917-4360